Refine Search

Search Results -

Terms	Documents
L1 and licens$3 near3 server	5

Database:
US Pre-Grant Publication Full-Text Database
US Patents Full-Text Database
US OCR Full-Text Database
EPO Abstracts Database
JPO Abstracts Database
Derwent World Patents Index
IBM Technical Disclosure Bulletins

Search: L2

Refine Search | Recall Text | Clear | Interrupt

Search History

DATE: Friday, November 04, 2005 Printable Copy Create Case

Set Name side by side	Query	Hit Count	Set Name result set
DB=JPAB; PLUR=YES; OP=OR			
L2	L1 and licens$3 near3 server	5	L2
L1	(digital near3 licens$3)	44	L1

END OF SEARCH HISTORY

Hit List

First Hit | Clear | Generate Collection | Print | Fwd Refs | Bkwd Refs | Generate OACS

Search Results - Record(s) 1 through 5 of 5 returned.

☐ 1. Document ID: JP 2004272395 A

L2: Entry 1 of 5 File: JPAB Sep 30, 2004

PUB-NO: JP02004272395A
DOCUMENT-IDENTIFIER: JP 2004272395 A
TITLE: INFORMATION MANAGING SYSTEM AND METHOD, INFORMATION PROCESSOR AND INFORMATION PROCESSING METHOD, PROGRAM, INFORMATION MANAGING DEVICE AND METHOD AND PROGRAM

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Draw D

☐ 2. Document ID: JP 2004227042 A

L2: Entry 2 of 5 File: JPAB Aug 12, 2004

PUB-NO: JP02004227042A
DOCUMENT-IDENTIFIER: JP 2004227042 A
TITLE: LICENSE DISTRIBUTION SERVER, LICENSE RE-ISSUING DEVICE CONNECTED TO THE SAME AND STORE FRONT SERVER

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Draw D

☐ 3. Document ID: JP 2004046856 A

L2: Entry 3 of 5 File: JPAB Feb 12, 2004

PUB-NO: JP02004046856A
DOCUMENT-IDENTIFIER: JP 2004046856 A
TITLE: METHOD FOR OBTAINING DIGITAL LICENSE CORRESPONDING TO DIGITAL CONTENT

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Draw D

☐ 4. Document ID: JP 2003333507 A

L2: Entry 4 of 5 File: JPAB Nov 21, 2003

PUB-NO: JP02003333507A
DOCUMENT-IDENTIFIER: JP 2003333507 A
TITLE: RECEIVER AND COPYING CONTROL METHOD

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Draw D

☐ 5. Document ID: JP 2001325387 A

L2: Entry 5 of 5 File: JPAB Nov 22, 2001

PUB-NO: JP02001325387A
DOCUMENT-IDENTIFIER: JP 2001325387 A
TITLE: DATA GENERATING DEVICE, DATA MANAGEMENT SYSTEM, DATA GENERATING METHOD, AND DATA MANAGING METHOD

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Draw D

Clear | Generate Collection | Print | Fwd Refs | Bkwd Refs | Generate OACS

Terms	Documents
L1 and licens$3 near3 server	5

Display Format: TI Change Format

Previous Page Next Page Go to Doc#

Rendering (computer graphics)

From Wikipedia, the free encyclopedia.

Rendering is the process of generating an image from a model, by means of a software program. The model is a description of three dimensional objects in a strictly defined language or data structure. It would contain geometry, viewpoint, texture and lighting information. The image is a digital image / raster graphics image.

It is one of the major sub-topics of **3D computer graphics**, and in practice always connected to the others. In the 'graphics pipeline' it's the last major step, giving the final appearance to the models and animation. With the increasing sophistication of computer graphics since the 1970s onward, it has become a more distinct subject.

It has uses in: computer games, simulators, movies/tv special effects, and design visualisation. Each employing a different balance of features and techniques. As a product, a wide variety of renderers are available. some are integrated into larger modelling and animation packages, some are stand-alone, some are free open-source projects. On the inside, a renderer is a carefully engineered program, based on a selective mixture of disciplines related to: light physics, visual perception, mathematics, and software development.

In the case of 3D graphics, rendering is a slow, computationally intensive process (typically for movie creation), or supported by realtime 3D hardware accelerators in graphics cards (typically for 3D computer games). The term is by analogy with an "artist's rendering" of a scene.

Contents

Usage

When the pre-image (a wireframe sketch usually) is complete, rendering is used, which adds in Bitmap textures or Procedural textures, lights, bump mapping, and relative position to other objects. The result is a completed image the consumer or intended viewer sees.

For movie animations, several images (frames) must be rendered, and stitched together in a program capable of making an animation of this sort. Most 3-D image editing programs can do this.

Features

A rendered image can be understood in terms of a number of visible features. Rendering research and development has been largely motivated by finding ways to simulate these efficiently. Some relate directly to particular algorithms and techniques, while others are produced together.

- **shading** — how the color and brightness of a surface varies with lighting
- **texture-mapping** — a method of applying detail to surfaces
- **bump-mapping** — a method of simulating small-scale bumpiness on surfaces
- **fogging/participating medium** — how light dims when passing through non-clear atmosphere or air
- **shadows** — the effect of obstructing light
- **soft shadows** — varying darkness caused by partially obscured light sources
- **reflection** — mirror-like or highly glossy reflection
- **transparency** — sharp transmission of light through solid objects
- **translucency** — highly scattered transmission of light through solid objects
- **refraction** — bending of light associated with transparency
- **indirect illumination** — surfaces illuminated by light reflected off other surfaces, rather than directly from a light source
- **caustics** (a form of indirect illumination) — reflection of light off a shiny object, or focusing of light through a transparent object, to produce bright highlights on another object
- **depth of field** — objects appear blurry or out of focus when too far in front of or behind the object in focus
- **motion blur** — objects appear blurry due to high-speed motion, or the motion of the camera
- **non-photorealistic rendering** — rendering of scenes in an artistic style, intended to look like a painting or drawing

Techniques

Two families of overall, light transport, techniques have emerged: **radiosity** — related to finite element mathematics, and **ray tracing** — related to monte carlo mathematics. Both can provide a framework for a fairly complete solution to the rendering equation. Such approaches can be very slow and computationally-intensive.

For real-time, a complete calculation is not currently possible. Much faster is to simplify with one or both of these common approximations: No illumination, just **texture mapping** — since the intrinsic colors of an object has the greatest influence on its appearance. Or **direct illumination** — light from light-source to surface, then reflected from surface to camera/eye, since this light path is usually dominant in a scene. These would often be augmented with other special-case effects, or precalculations.

There are two large-scale approaches to rendering that can be applied to almost any rendering task. Which approach is chosen for a given task greatly influences the kind of problems that will have to be solved to create a successful renderer.

Pixel-by-pixel

As stated earlier, rendering is the problem of deciding what color each pixel should be, given a high-level representation of an image. One way to accomplish rendering is to take this description literally: loop over each pixel in the image, and determine based on the high-level description what color should be assigned to that pixel.

This approach is frequently used when one is rendering to perform color balancing, video compositing, and similar effects. In this case, each pixel in the rendered image depends on exactly one pixel in each of the source images, so a pixel-by-pixel technique is the natural approach.

When this approach is applied to 3D rendering, one arrives at the most basic version of the ray tracing algorithm.

Primitive-by-primitive

A high-level representation of an image necessarily contains elements in a different domain from pixels. These elements are referred to as primitives. In a schematic drawing, for instance, line segments and curves might be primitives. In a graphical user interface, windows and buttons might be the primitives. In 3D rendering, triangles and polygons in space might be primitives.

If a pixel-by-pixel approach to rendering is impractical or too slow for some task, then a primitive-by-primitive approach to rendering may prove useful. Here, one loops through each of the primitives, determines which pixels in the image it affects, and modifies those pixels accordingly. This is called **rasterization**, and is the rendering method used by all current graphics cards.

Rasterization is frequently faster than pixel-by-pixel rendering. First, large areas of the image may be empty of primitives; rasterization will ignore these areas, but pixel-by-pixel rendering must pass through them. Second, rasterization can improve cache coherency and reduce redundant work by taking advantage of the fact that the pixels occupied by a single primitive tend to be contiguous in the image. For these reasons, rasterization is usually the approach of choice when interactive rendering is required; however, the pixel-by-pixel approach can often produce higher-quality images and is more versatile because it does not depend on as many assumptions about the image as rasterization.

Sampling and filtering

One problem that any rendering system must deal with, no matter which approach it takes, is the **sampling problem**. Essentially, the rendering process tries to depict a continuous function from image space to colors by using a finite number of pixels. As a consequence of the Nyquist theorem, the scanning frequency must be twice the dot rate, which is proportional to image resolution. In simpler terms, this expresses the idea that an image cannot display details smaller than one pixel.

If a naive rendering algorithm is used, high frequencies in the image function will cause ugly aliasing to be present in the final image. Aliasing typically manifests itself as jaggies, or jagged edges on objects where the pixel grid is visible. In order to remove aliasing, all rendering algorithms (if they are to produce good-looking images) must filter the image function to remove high frequencies, a process called antialiasing.

- the painter's algorithm
- Scanline algorithms like Reyes
- Z-buffer algorithms
- Global illumination
- Radiosity
- Ray tracing
- Volume rendering

Rendering for movies often takes place on a network of tightly connected computers known as a render farm.

The current state of the art in 3-D image description for movie creation is the RenderMan scene description language designed at Pixar. (compare with simpler 3D fileformats such as VRML or APIs such as OpenGL and DirectX tailored for 3D hardware accelerators).

Movie type rendering software includes:

- RenderMan compliant renderers
- Mental Ray
- Brazil
- Blender (may also be used for modeling)
- Gelato (http://film.nvidia.com/page/gelato.html)

Academic core

Most rendering development and use aims at **photorealism** — to produce images indistinguishable from photographs.

The implementation of a realistic renderer always has some basic element of physical simulation or emulation — some computation which resembles or abstracts a real physical process.

The term "*physically-based*" indicates the use of physical models and approximations that are more general and widely accepted outside rendering. A particular set of related techniques have gradually become established in the rendering community.

The basic concepts are moderately straightforward, but intractable to calculate; and a single elegant algorithm or approach has been elusive for more general purpose renderers. In order to meet demands of robustness, accuracy, and practicality, an implementation will be a complex combination of different techniques.

Rendering research is concerned with both the adaptation of scientific models and their efficient application.

The rendering equation

Main article: Rendering equation

This is the key academic/theoretical concept in rendering. It serves as the most abstract formal expression of the non-perceptual aspect of rendering. All more complete algorithms can be seen as solutions to particular formulations of this equation.

$$L_o(x, \vec{w}) = L_e(x, \vec{w}) + \int_\Omega f_r(x, \vec{w}', \vec{w}) L_i(x, \vec{w}')(\vec{w}' \cdot \vec{n}) d\vec{w}'$$

Meaning: at a particular position and direction, the outgoing light (L_o) is the sum of the emitted light (L_e) and the reflected light. The reflected light being the sum of the incoming light (L_i) from all directions, multiplied by the surface reflection and incoming angle. By connecting outward light to inward light, via an interaction point, this equation stands for the whole 'light transport' in a scene.

The Bidirectional Reflectance Distribution Function

The **Bidirectional Reflectance Distribution Function** (BRDF) expresses a simple model of light interaction with a surface as follows:

$$f_r(x, \vec{w}', \vec{w}) = \frac{dL_r(x, \vec{w})}{L_i(x, \vec{w}')(\vec{w}' \cdot \vec{n}) d\vec{w}'}$$

Light interaction is often approximated by the even simpler models: diffuse reflection and specular reflection,

although both can be BRDFs.

Geometric optics

Rendering is practically exclusively concerned with the particle aspect of light physics — known as geometric optics. Treating light, at its basic level, as particles bouncing around is a simplification, but appropriate: the wave aspects of light are negligible in most scenes, and are significantly more difficult to simulate. Notable wave aspect phenomena include diffraction — as seen in the colours of CDs and DVDs — and polarisation — as seen in LCDs. Both types of effect, if needed, are made by appearance-oriented adjustment of the reflection model.

Visual perception

Though it receives less attention, an understanding of human visual perception is valuable to rendering. This has two causes: image displays have restricted ranges, and human perception has restricted ranges. A renderer can simulate an almost infinite range of light brightness and color, but current displays — movie screen, computer monitor, etc. — cannot handle so much, and something must be discarded or 'compressed'. Human perception also has limits, and so doesn't need to be given large-range images to create realism. This can help solve the problem of fitting images into displays, and, furthermore, suggest what short-cuts could be used in the rendering simulation, since certain subtleties won't be noticeable.

Mathematics used in rendering includes: linear algebra, calculus, numerical mathematics, signal processing, monte carlo.

Chronology of important published ideas

- 1970 **Scan-line algorithm** (Bouknight, W. J. (1970). A procedure for generation of three-dimensional half-tone computer graphics presentations. *Communications of the ACM*)
- 1971 **Gouraud shading** (Gouraud, H. (1971). Computer display of curved surfaces. *IEEE Transactions on Computers* **20** (6), 623–629.)
- 1974 **Texture mapping** (Catmull, E. (1974). A subdivision algorithm for computer display of curved surfaces. *PhD thesis*, University of Utah.)
- 1974 **Z-buffer** (Catmull, E. (1974). A subdivision algorithm for computer display of curved surfaces. *PhD thesis*)
- 1975 **Phong shading** (Phong, B-T. (1975). Illumination for computer generated pictures. *Communications of the ACM* **18** (6), 311–316.)
- 1976 **Environment mapping** (Blinn, J.F. Newell, M.E. (1976). Texture and reflection in computer generated images. *Communications of the ACM* **19**, 542–546.)
- 1977 **Shadow volumes** (Crow, F.C. (1977). Shadow algorithms for computer graphics. *Computer Graphics (Proceedings of SIGGRAPH 1977)* **11** (2), 242–248.)
- 1978 **Shadow buffer** (Williams, L. (1978). Casting curved shadows on curved surfaces. *Computer Graphics (Proceedings of SIGGRAPH 1978)* **12** (3), 270–274.)
- 1978 **Bump mapping** (Blinn, J.F. (1978). Simulation of wrinkled surfaces. *Computer Graphics (Proceedings of SIGGRAPH 1978)* **12** (3), 286–292.)
- 1980 **BSP trees** (Fuchs, H. Kedem, Z.M. Naylor, B.F. (1980). On visible surface generation by a priori tree structures. *Computer Graphics (Proceedings of SIGGRAPH 1980)* **14** (3), 124–133.)
- 1980 **Ray tracing** (Whitted, T. (1980). An improved illumination model for shaded display. *Communications of the ACM* **23** (6), 343–349.)
- 1981 **Cook shader** (Cook, R.L. Torrance, K.E. (1981). A reflectance model for computer graphics. *Computer Graphics (Proceedings of SIGGRAPH 1981)* **15** (3), 307–316.)
- 1983 **Mipmaps** (Williams, L. (1983). Pyramidal parametrics. *Computer Graphics (Proceedings of SIGGRAPH 1983)* **17** (3), 1–11.)
- 1984 **Octree ray tracing** (Glassner, A.S. (1984). Space subdivision for fast ray tracing. *IEEE Computer*

Graphics & Applications **4** (10), 15–22.)
- 1984 **Alpha compositing** (Porter, T. Duff, T. (1984). Compositing digital images. *Computer Graphics (Proceedings of SIGGRAPH 1984)* **18** (3), 253–259.)
- 1984 **Distributed ray tracing** (Cook, R.L. Porter, T. Carpenter, L. (1984). Distributed ray tracing. *Computer Graphics (Proceedings of SIGGRAPH 1984)* **18** (3), 137–145.)
- 1984 **Radiosity** (Goral, C. Torrance, K.E. Greenberg, D.P. Battaile, B. (1984). Modelling the interaction of light between diffuse surfaces. *Computer Graphics (Proceedings of SIGGRAPH 1984)* **18** (3), 213–222.)
- 1985 **Hemi-cube radiosity** (Cohen, M.F. Greenberg, D.P. (1985). The hemi-cube: a radiosity solution for complex environments. *Computer Graphics (Proceedings of SIGGRAPH 1985)* **19** (3), 31–40.)
- 1986 **Light source tracing** (Arvo, J. (1986). Backward ray tracing. *SIGGRAPH 1986 Developments in Ray Tracing course notes*)
- 1986 **Rendering equation** (Kajiya, J.T. (1986). The rendering equation. *Computer Graphics (Proceedings of SIGGRAPH 1986)* **20** (4), 143–150.)
- 1987 **Reyes algorithm** (Cook, R.L. Carpenter, L. Catmull, E. (1987). The reyes image rendering architecture. *Computer Graphics (Proceedings of SIGGRAPH 1987)* **21** (4), 95–102.)
- 1991 **Hierarchical radiosity** (Hanrahan, P. Salzman, D. Aupperle, L. (1991). A rapid hierarchical radiosity algorithm. *Computer Graphics (Proceedings of SIGGRAPH 1991)* **25** (4), 197–206.)
- 1993 **Tone mapping** (Tumblin, J. Rushmeier, H.E. (1993). Tone reproduction for realistic computer generated images. *IEEE Computer Graphics & Applications* **13** (6), 42–48.)
- 1993 **Subsurface scattering** (Hanrahan, P. Krueger, W. (1993). Reflection from layered surfaces due to subsurface scattering. *Computer Graphics (Proceedings of SIGGRAPH 1993)* **27** (), 165–174.)
- 1995 **Photon mapping** (Jensen, H.J. Christensen, N.J. (1995). Photon maps in bidirectional monte carlo ray tracing of complex objects. *Computers & Graphics* **19** (2), 215–224.)

Books and summaries

- Foley; Van Dam; Feiner; Hughes (1990). *Computer Graphics: Principles And Practice*. Addison Wesley. ISBN 0201121107.
- Glassner (1995). *Principles Of Digital Image Synthesis*. Morgan Kaufmann. ISBN 1558602763.
- Pharr; Humphreys (2004). *Physically Based Rendering*. Morgan Kaufmann. ISBN 012553180X.
- Dutre; Bala; Bekaert (2002). *Advanced Global Illumination*. AK Peters. ISBN 1568811772.
- Jensen (2001). *Realistic Image Synthesis Using Photon Mapping*. AK Peters. ISBN 1568811470.
- Shirley; Morley (2003). *Realistic Ray Tracing* (2nd ed.). AK Peters. ISBN 1568811985.
- Glassner (1989). *An Introduction To Ray Tracing*. Academic Press. ISBN 0122861604.
- Cohen; Wallace (1993). *Radiosity and Realistic Image Synthesis*. AP Professional. ISBN 0121782700.
- Akenine-Moller; Haines (2002). *Real-time Rendering* (2nd ed.). AK Peters. ISBN 1568811829.
- Gooch; Gooch (2001). *Non-Photorealistic Rendering*. AKPeters. ISBN 1568811330.
- Strothotte; Schlechtweg (2002). *Non-Photorealistic Computer Graphics*. Morgan Kaufmann. ISBN 1558607870.
- Blinn (1996). *Jim Blinns Corner - A Trip Down The Graphics Pipeline*. Morgan Kaufmann. ISBN 1558603875.
- Description of the 'Radiance' system (http://radsite.lbl.gov/radiance/papers/sg94.1/)

External links

- SIGGRAPH (http://www.siggraph.org/) The ACMs special interest group in graphics — the largest academic and professional association and conference.
- Gelato (http://film.nvidia.com/page/gelato.html) hardware-accelerated "no compromises" film-quality renderer from NVIDIA
- Ray Tracing News (http://www.raytracingnews.org/) A newsletter on ray tracing technical matters.
- Real-Time Rendering resources (http://www.realtimerendering.com/) A list of links to resources, associated with the *Real-Time Rendering* book.
- http://www.graphicspapers.com/ Database of graphics papers citations.

- http://www.cs.brown.edu/~tor/ List of links to (recent) siggraph papers (and some others) on the web.
- http://www.pointzero.nl/renderers/ List of links to all kinds of renderers.
- http://www.renderman.org/
- 'Radiance' renderer. (http://radsite.lbl.gov/radiance/) A highly accurate ray-tracing software system.
- 'Aqsis' renderer (http://www.aqsis.org/) A free RenderMan compatible OpenSource REYES renderer.
- 'GenesisIV' renderer (http://www.geomantics.com/) A photorealistic landscape renderer.
- http://www.povray.org/ A free ray tracer.
- 3D Computer Graphics and Visual Arts at Xmeta.com (http://www.xmeta.com/web/422293/arts/visual-arts/computer-graphics/3d/) A directory of 3D graphics resources, including artists, tutorials, products, etc.

Retrieved from "http://en.wikipedia.org/wiki/Rendering_%28computer_graphics%29"

Categories: Computer graphics | 3D computer graphics